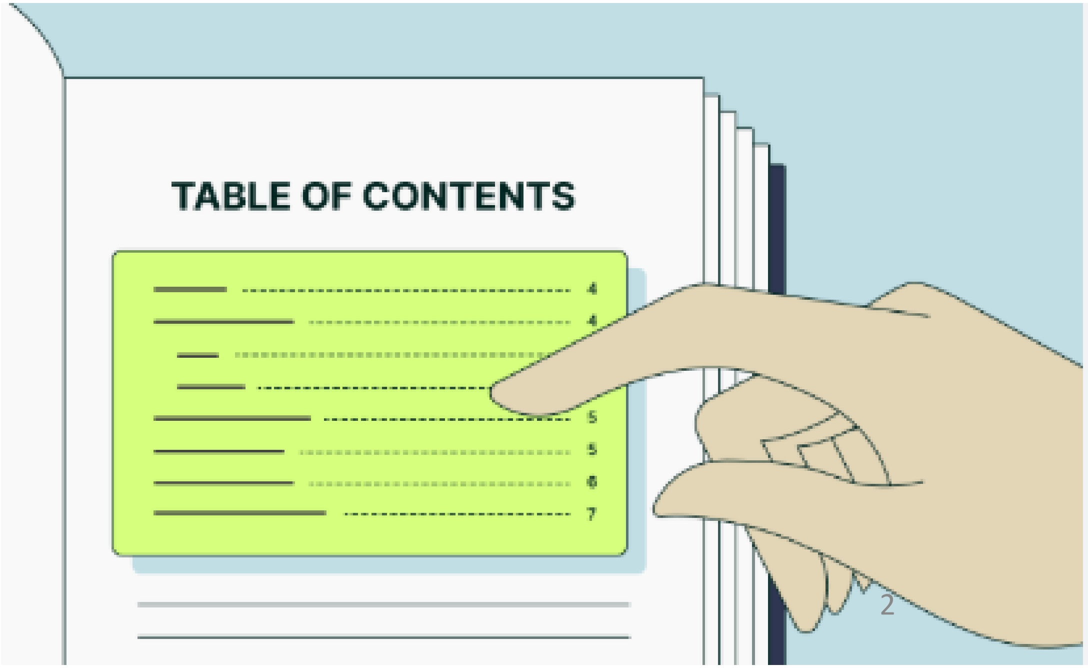
Unit-1

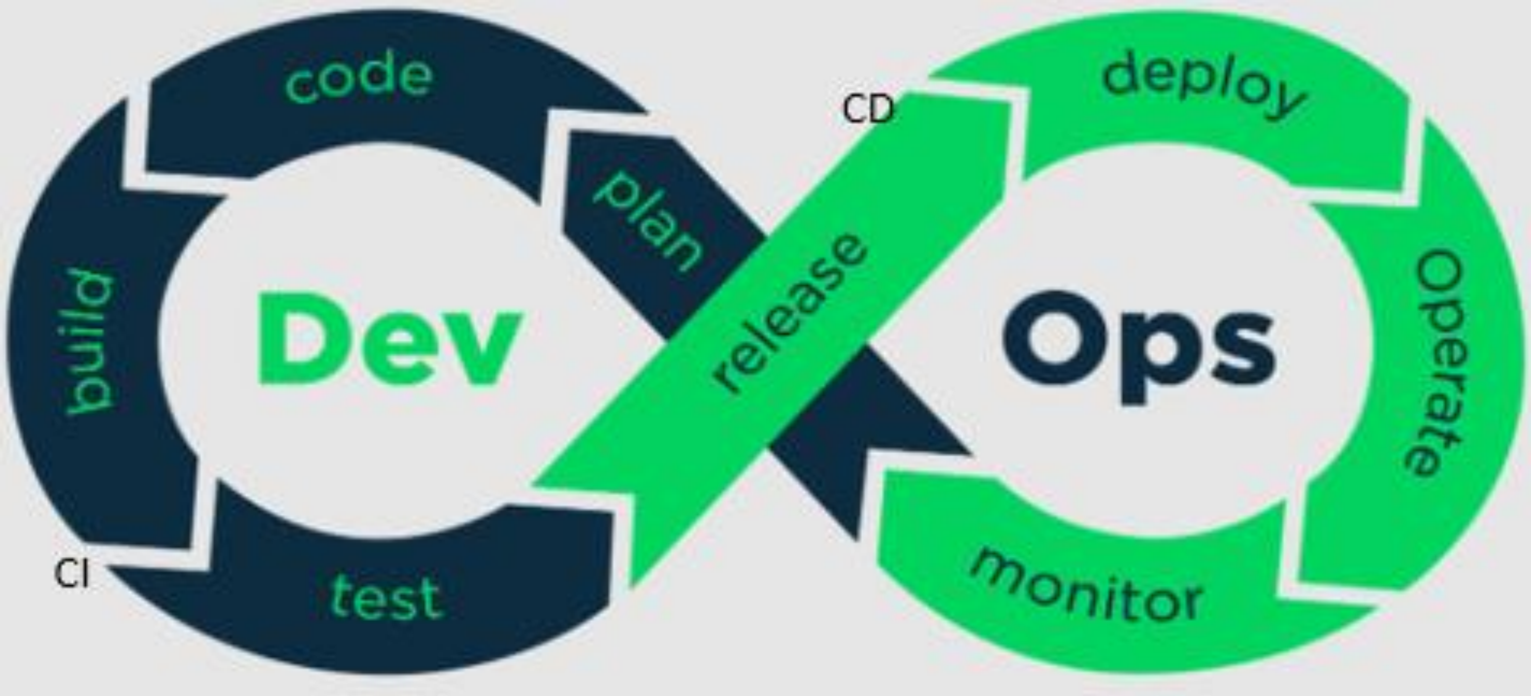
Introduction to S/w Engineering

Table of Content

- What is DevOps?
- DevOps Collaboration Cycle
- Meaning of infinite loop
- Benefits of DevOps
- DevOps Tools
- CI/CD Pipeline
- Benefits of CI/CD



Process


code
plan
build
Dev
test
CI
CD
release
deploy
Ops
operate
monitor

What is DevOps?

Tool

Software

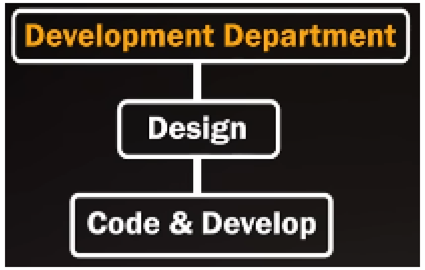
Software Development
DEVOPS

Feedback Loop

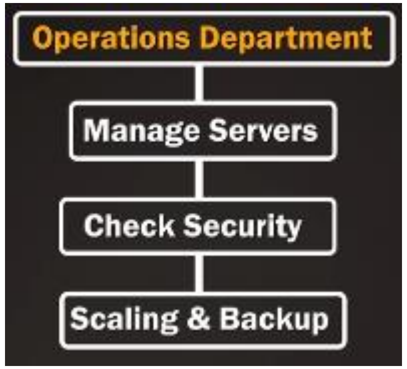
Product Development <- **Application Release**

Development Department

Operations Department

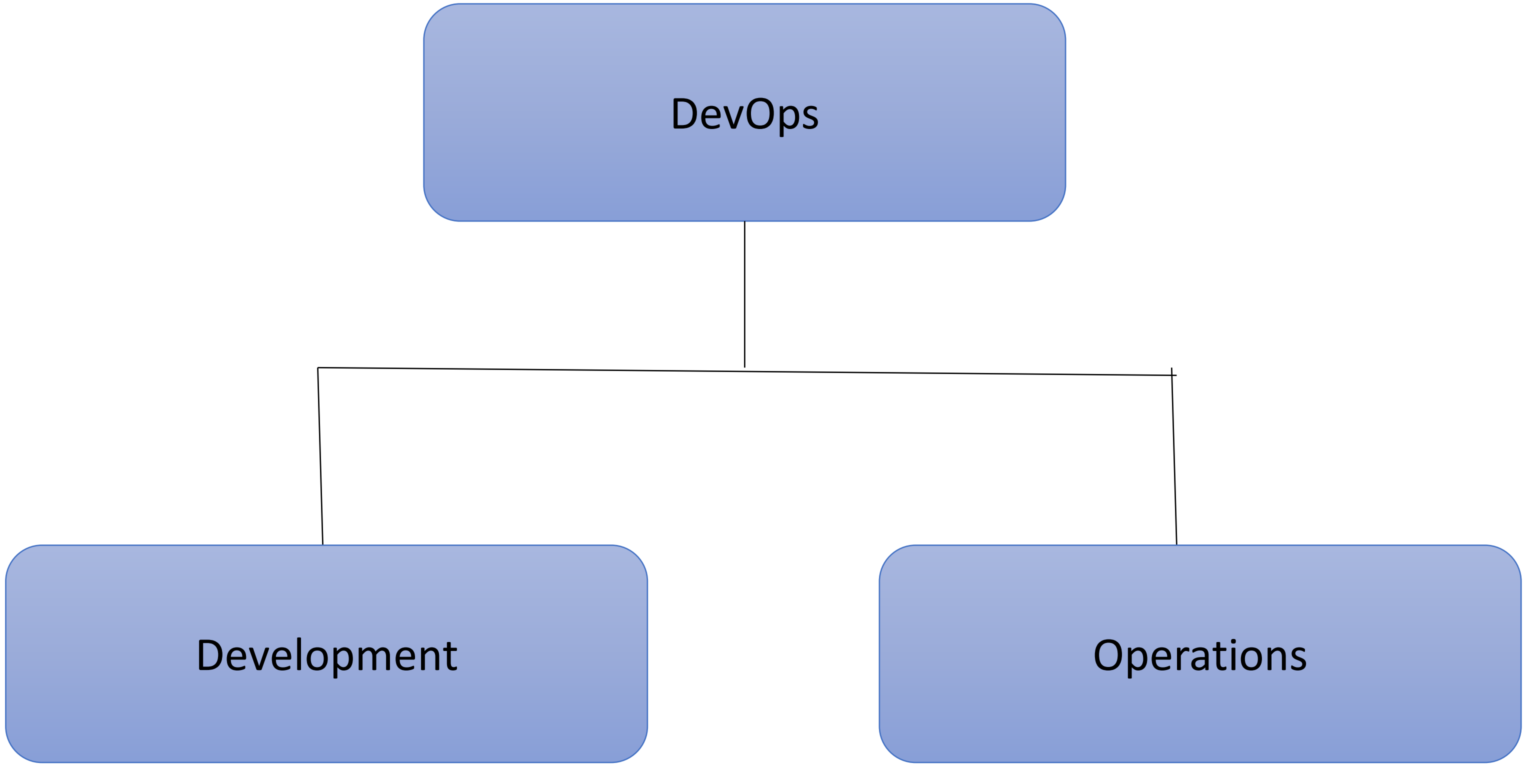


Development Department
Design
Code & Develop



Operations Department
Manage Servers
Check Security
Scaling & Backup

DevOps



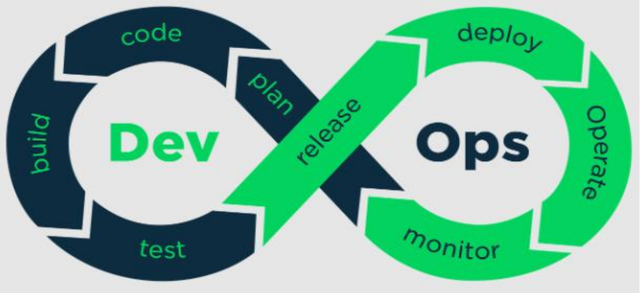
DevOps is a set of practices and tools designed **to shorten the life cycle** of a software development process

What is DevOps?

- DevOps is a s/w development approach that emphasizes **collaboration and communication** ***between development and operations team.***

- Allows a single team to ***handle the entire application lifecycle*** from development to testing, deployment and operations

- Reduce the disconnection between **software developers**, **quality assurance engineers** and **system administrators**

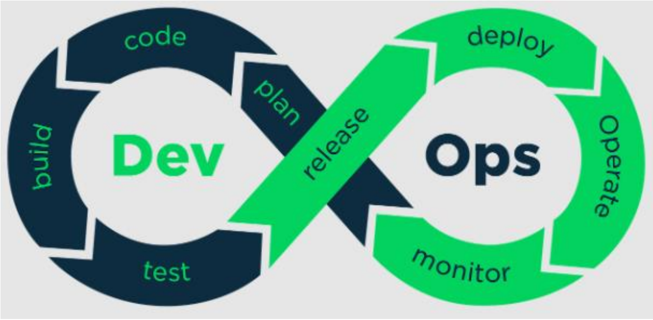


What is DevOps?

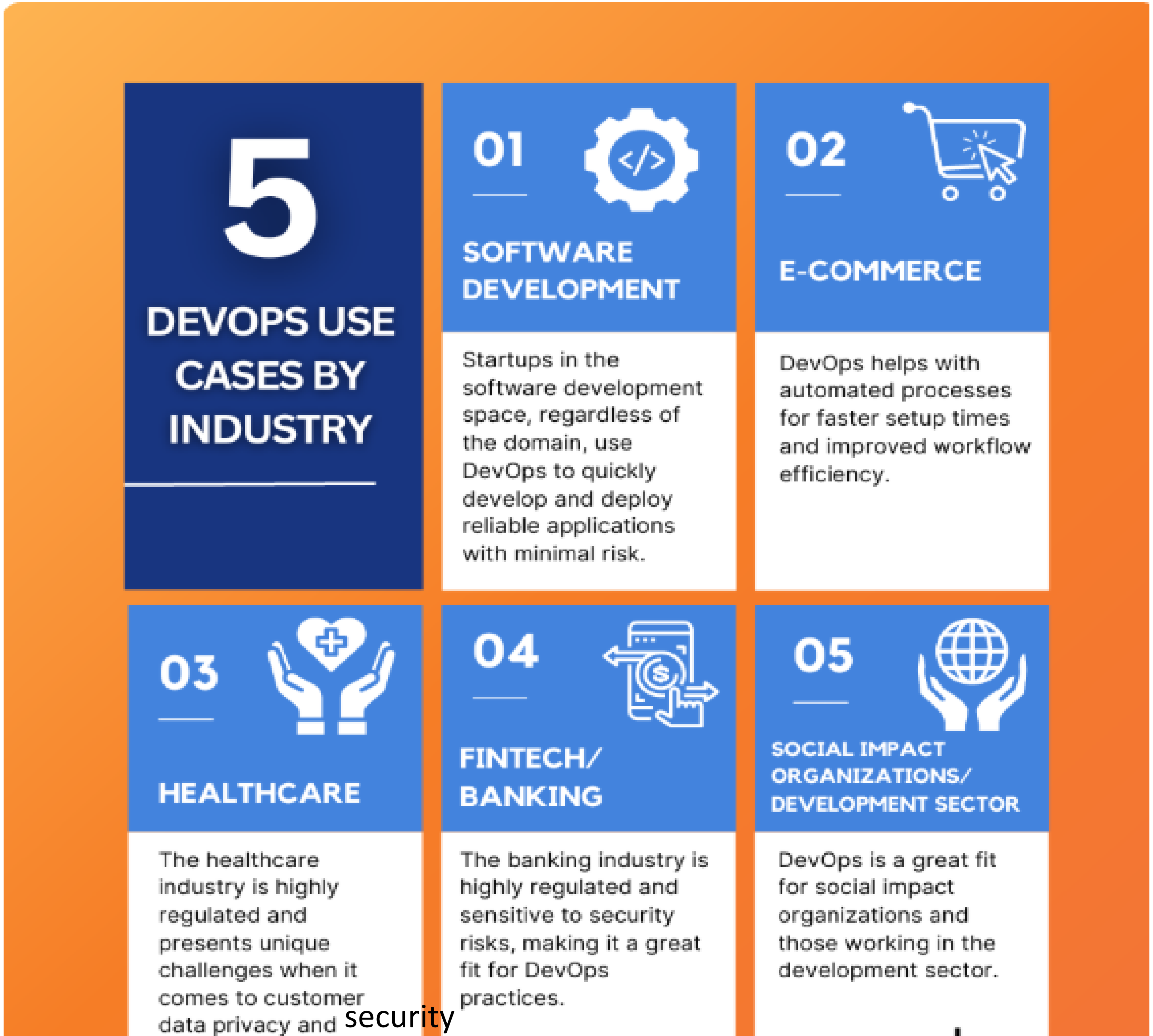
Collaboration between Software Developers and IT Operations Team.

Emphasizes automation, communication, fast feedback, and iterative improvement to accelerate software delivery.

Enables organizations to deliver software faster, with higher quality, and more reliably.



How DevOps practices are applied across different industries?



5 DEVOPS USE CASES BY INDUSTRY

01 SOFTWARE DEVELOPMENT

Startups in the software development space, regardless of the domain, use DevOps to quickly develop and deploy reliable applications with minimal risk.

02 E-COMMERCE

DevOps helps with automated processes for faster setup times and improved workflow efficiency.

03 HEALTHCARE

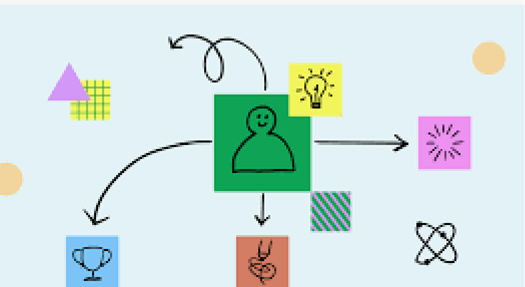
The healthcare industry is highly regulated and presents unique challenges when it comes to customer data privacy and security

04 FINTECH/ BANKING

The banking industry is highly regulated and sensitive to security risks, making it a great fit for DevOps practices.

05 SOCIAL IMPACT ORGANIZATIONS/ DEVELOPMENT SECTOR

DevOps is a great fit for social impact organizations and those working in the development sector.



Contd.

1. Software Development

- **Description: Startups and companies** in software development adopt DevOps to build and deploy reliable applications efficiently.
- **Key Benefits**:
  - Accelerated development and deployment cycles.
  - Reduction in risks associated with releasing software.

2. E-Commerce

- **Description**: E-commerce businesses leverage DevOps for faster setup times and optimized workflows.
- **Key Benefits**:
  - Automated processes streamline operations.
  - Enhances workflow efficiency, reducing time-to-market for features.

Contd.

3. Healthcare

- **Description**: The healthcare industry, being heavily regulated, relies on DevOps for addressing data privacy and security challenges.
- **Key Benefits**:
  - Compliance with strict regulations.
  - Improved security and reliability in handling sensitive patient data.

4. Fintech/Banking

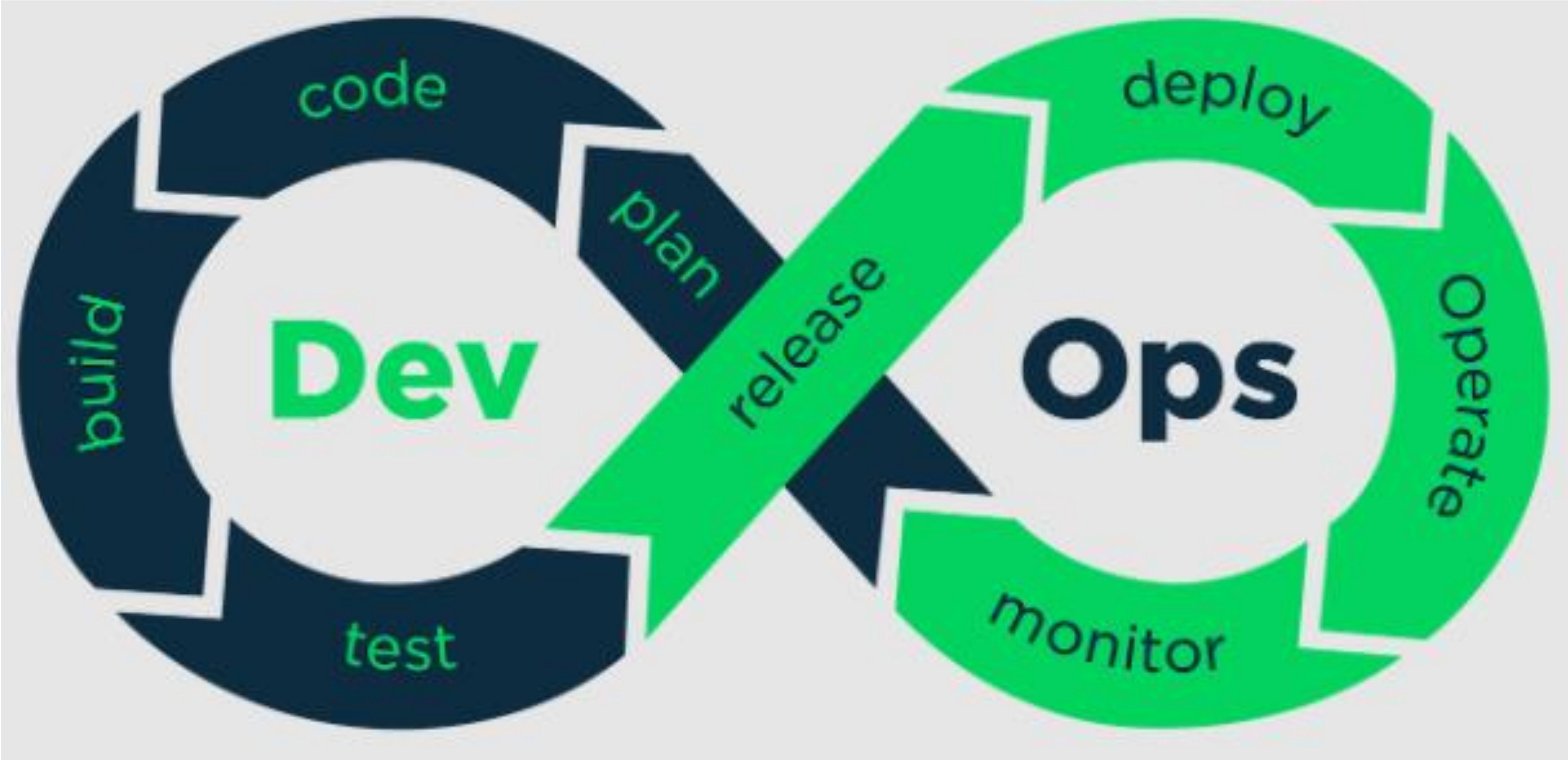
- **Description**: In the highly sensitive and regulated banking sector, DevOps is used to address security concerns while maintaining agility.
- **Key Benefits**:
  - Enhanced security protocols.
  - Ensures compliance with industry regulations.
  - Facilitates continuous delivery while minimizing risks.

Contd.

5. Social Impact Organizations/Development Sector

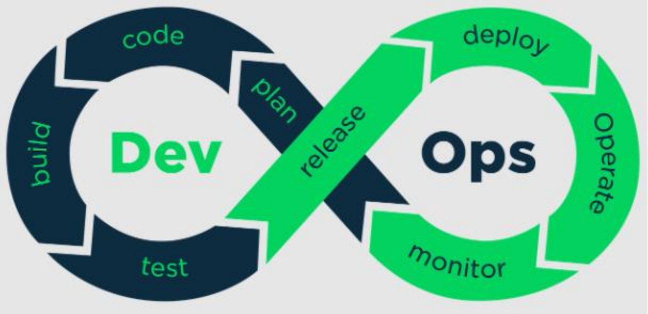
- **Description**: DevOps supports social impact organizations by enabling them to achieve efficiency and scalability in their operations.
- **Key Benefits**:
  - Accelerates project timelines.
  - Ensures reliable deployment of applications in impactful development initiatives.

DevOps collaboration cycle:



DevOps cycle

Meaning of infinite loop:



illustrates how **development and operations teams** work together in a never-ending process to build, deploy, operate and improve applications.

Continuous Feedback:

Seamless feedback between **development and operations at every stage** ensures *continuous improvement of software and processes*.

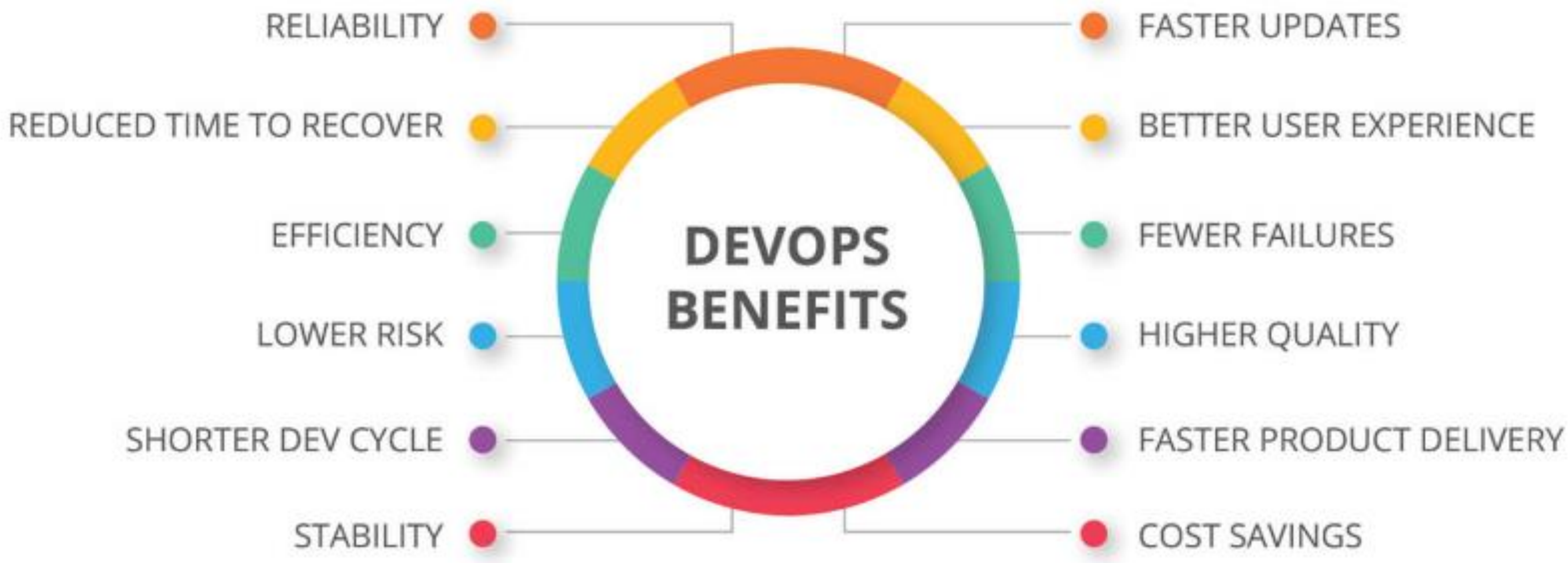
Seamless Collaboration:

Reflects the integration of teams,(teams work together smoothly without barriers) breaking down silos between development and operations.

Automation and Iteration:

Processes like **testing, deployment, and monitoring** are automated and repeat in cycles which reduces manual effort and increases efficiency.

Benefits of DevOps:



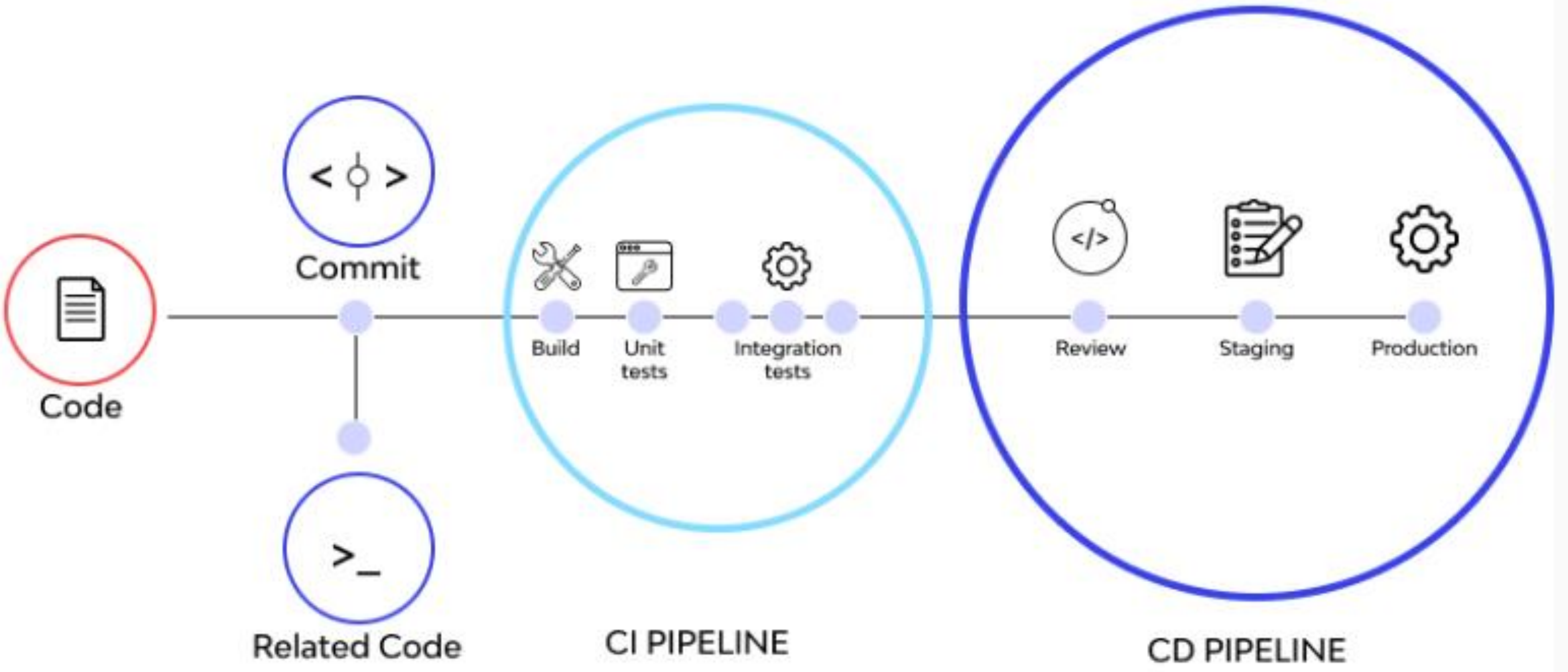
[DevOps benefits]

DevOps Tools:

- **Version Control:** Git, GitHub, GitLab
- **CI/CD Tools:** Jenkins, GitLab CI/CD
- **Configuration Management:** Ansible, Puppet, Chef
- **Containerization:** Docker, Kubernetes
- **Monitoring:** Prometheus, ELK Stack

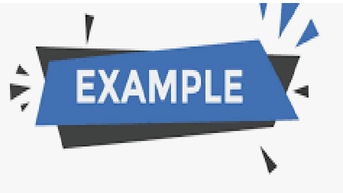
CI/CD (Continuous Integration and Continuous Deployment) pipeline

A **process used** in <u>software development</u> **to automate**
building, testing and deploying code.



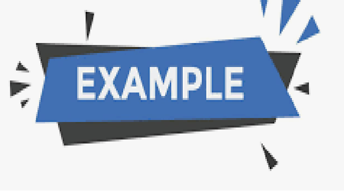
Imagine you are developing a food delivery app like **Uber Eats**

- **Code**: A **developer adds a feature**, like a new "dark mode" for the app.



- **Commit**: The **developer saves** (commits) the changes to a shared repository (like GitHub).
- **CI Pipeline**:
  - **Build**: The system automatically builds the app with the new feature to check if it ***compiles*** correctly.
  - **Unit Tests**: It tests only the "dark mode" feature to ensure it works as expected.
  - **Integration Tests**: It ensures the new feature works with existing app features, **like** the ***checkout or order screens.***

Contd.



- **CD Pipeline**:
  - **Review**: The team reviews the feature for **quality** (optional for Continuous Deployment).
  - **Staging**: The **app is deployed** to a staging environment where a team or beta users **test** "dark mode" as if it's live.
  - **Production**: Once stable, the feature is automatically or manually deployed to the app that users download from app stores.

Real-Life Benefit: Every small change, like "dark mode," is tested, integrated and delivered seamlessly to users with minimal downtime or errors.

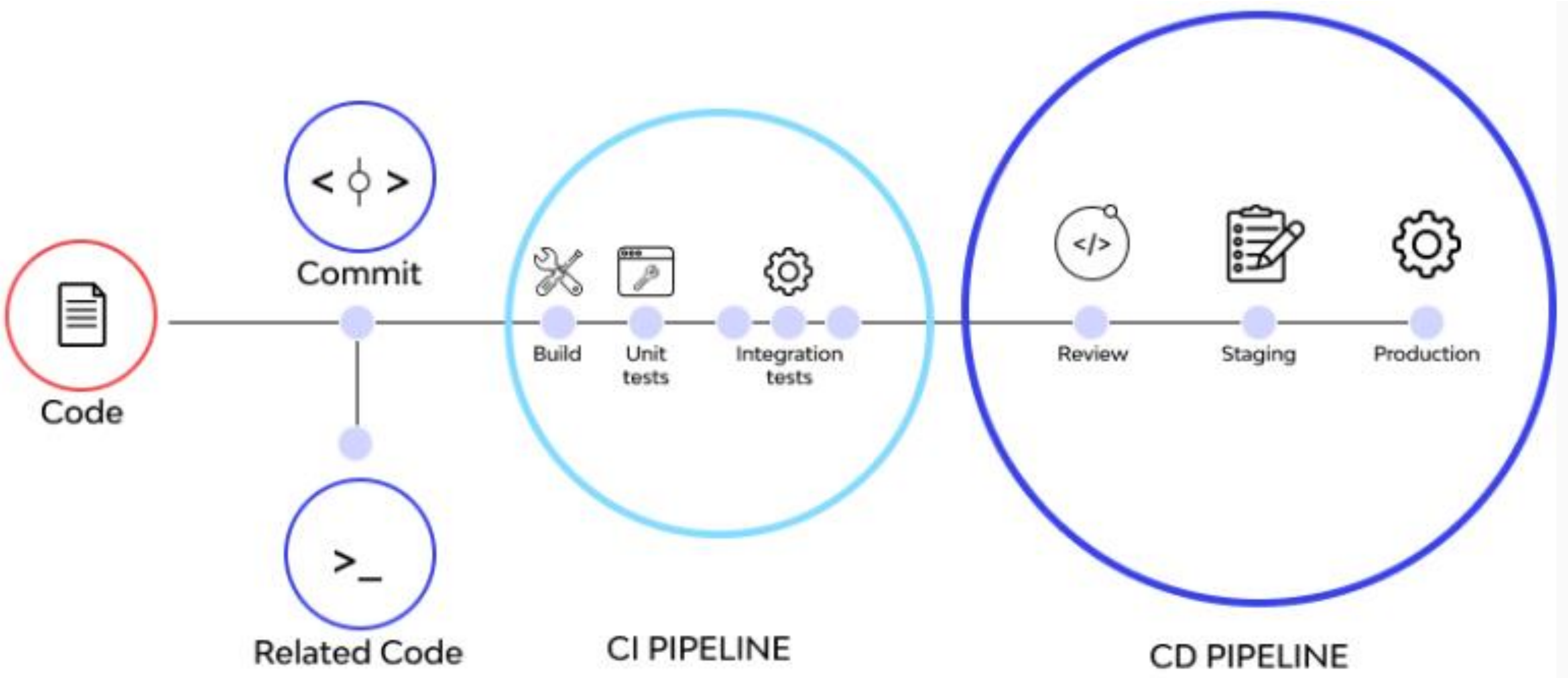
CI/CD Pipeline

CI: Continuous Integration **CD:** Continuous Delivery/Deployment

- Enabling Continuous Integration and Deployment **for Modern software**
- CI/CD pipelines enable **faster and reliable** software delivery processes.
- It is a software development approach **to automate the process of** integrating code changes, testing, and deploying applications.
- **Automates** builds, tests and deployments for seamless collaboration.
- **Tools like** Jenkins, Docker, Kubernetes streamline pipeline workflows.

What is CI/CD?

- **Continuous Integration**: Merging code into shared repository frequently.
- **Continuous Deployment:** Automating software delivery to production environments.
- Goal: Streamline development, testing, and deployment processes efficiently.



Tools for CI/CD

- **Version Control**: *Git, GitHub, GitLab* for source management.
- **Build Tools**: *Jenkins, CircleCI, TravisCI* automate build processes**.**
- **Containerization**: *Docker* simplifies deploying applications using containers.
- **Orchestration**: *Kubernetes* manages containerized applications at scale.
- **Testing Tools**: *Selenium, JUnit* ensure automated software testing workflows.

CI/CD Pipeline Workflow

- **Source Control**: Code pushed to version control repository.
- **Build Stage**: Application code compiled and dependencies installed.
- **Test Stage**: Automated tests run to ensure software quality.
- **Release Stage**: Deployment package prepared and versioned for delivery.
- **Deploy Stage**: Application deployed to production or staging environments.
- **Monitor Stage**: Application performance monitored post-deployment.

Benefits of CI/CD:

- **Faster Time-to-Market**: Accelerates delivery of features and updates.
- **Improved Code Quality**: Automated testing reduces bugs and regression issues.
- **Early Error Detection**: Integration issues are caught earlier.
- **Reliable Deployment**: Automation reduces manual errors during release.
- **Enhanced Collaboration**: Encourages frequent and smaller code changes.

Challenges in CI/CD Adoption

1. **Cultural Resistance**: Teams may resist transitioning from traditional workflows.
2. **Toolchain Complexity**: Integrating and managing CI/CD tools is challenging.
3. **Skill Gaps**: Teams may lack expertise in CI/CD processes and tools.
4. **Infrastructure Costs**: Setting up and maintaining CI/CD can be expensive.
5. **Testing Challenges**: Comprehensive test coverage is time-consuming to build and maintain.
6. **Security Concerns**: Automated pipelines can expose vulnerabilities if not secure.
7. **Integration Issues**: Legacy systems may not integrate well with CI/CD pipelines.
8. **Change Management**: Frequent changes can lead to instability if not managed properly.
9. **Monitoring and Feedback**: Failures in pipelines can go unnoticed without robust monitoring.
10. **Scalability**: Scaling pipelines for large teams and complex apps is difficult.

Summary of workflow

Diagram illustrates a **DevOps pipeline** integrating tools and platforms like Git, Jenkins, Docker, and Kubernetes.

- **Source Code Management**: Developers push code to GitHub.
- **Continuous Integration**: Jenkins builds and tests the code.
- **Containerization**: Docker packages the application into images and uploads them to Docker Hub.
- **Continuous Deployment**: Kubernetes manages the deployment and scaling of the application in production.

pipeline demonstrates a modern DevOps process that enables **automation, scalability, and efficiency** in deploying applications.

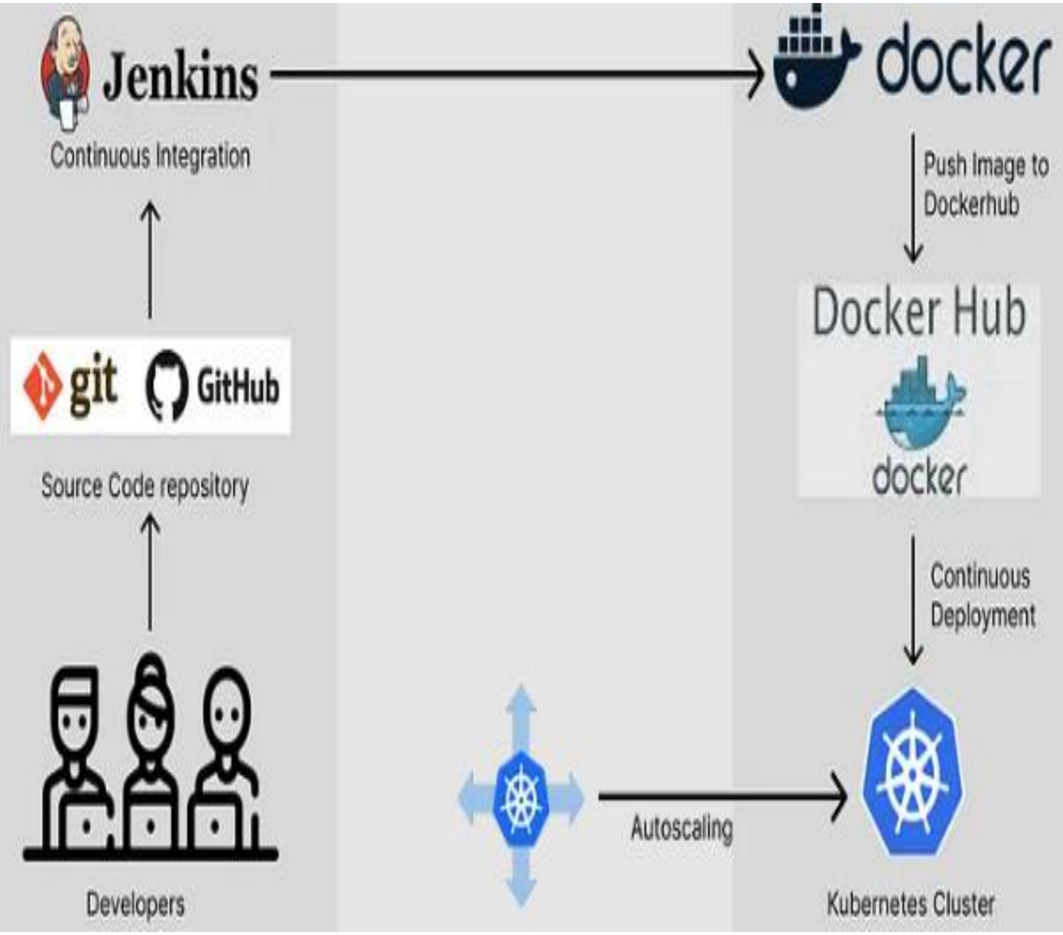


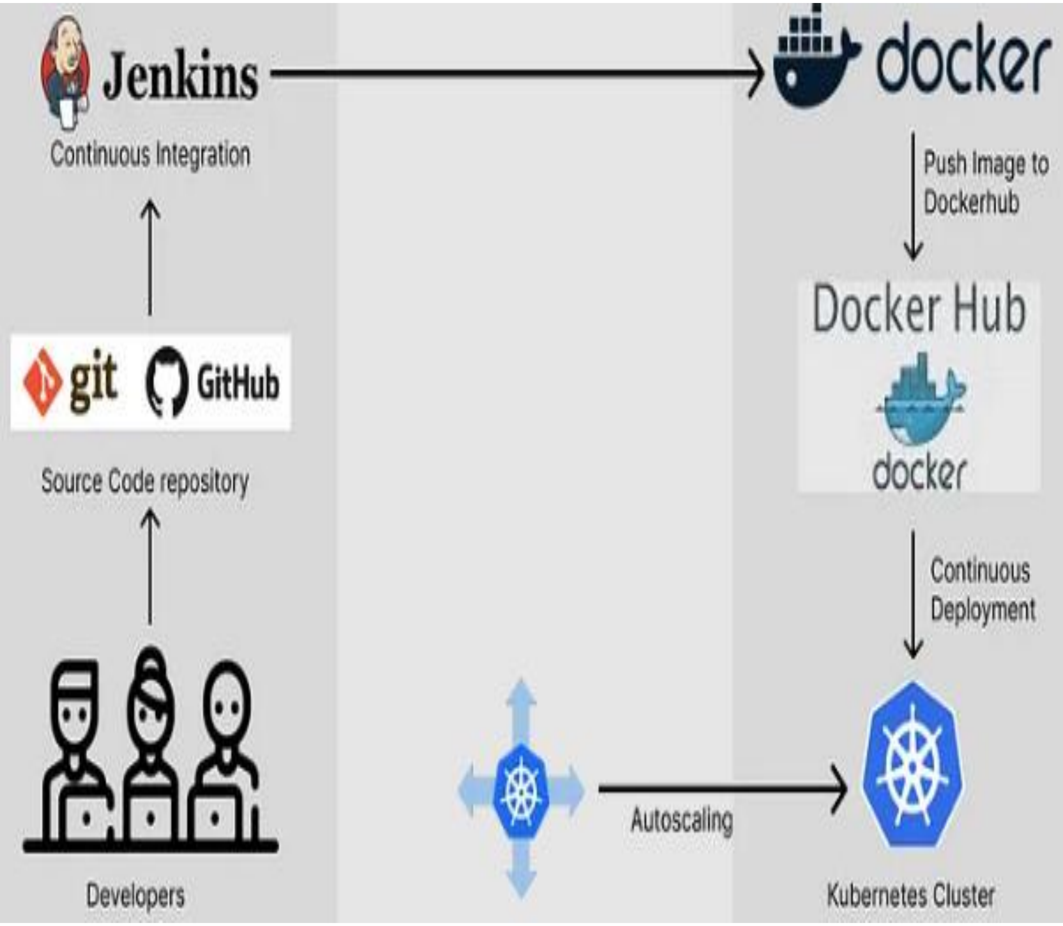
Diagram illustrates a **DevOps pipeline** integrating tools and platforms like Git, Jenkins, Docker, and Kubernetes.

1. Developers and Source Code Repository

- **Developers**: Write and commit source code.
- **Git/GitHub**: The source code is stored and version-controlled in Git or GitHub, which acts as the **central repository** for the codebase.

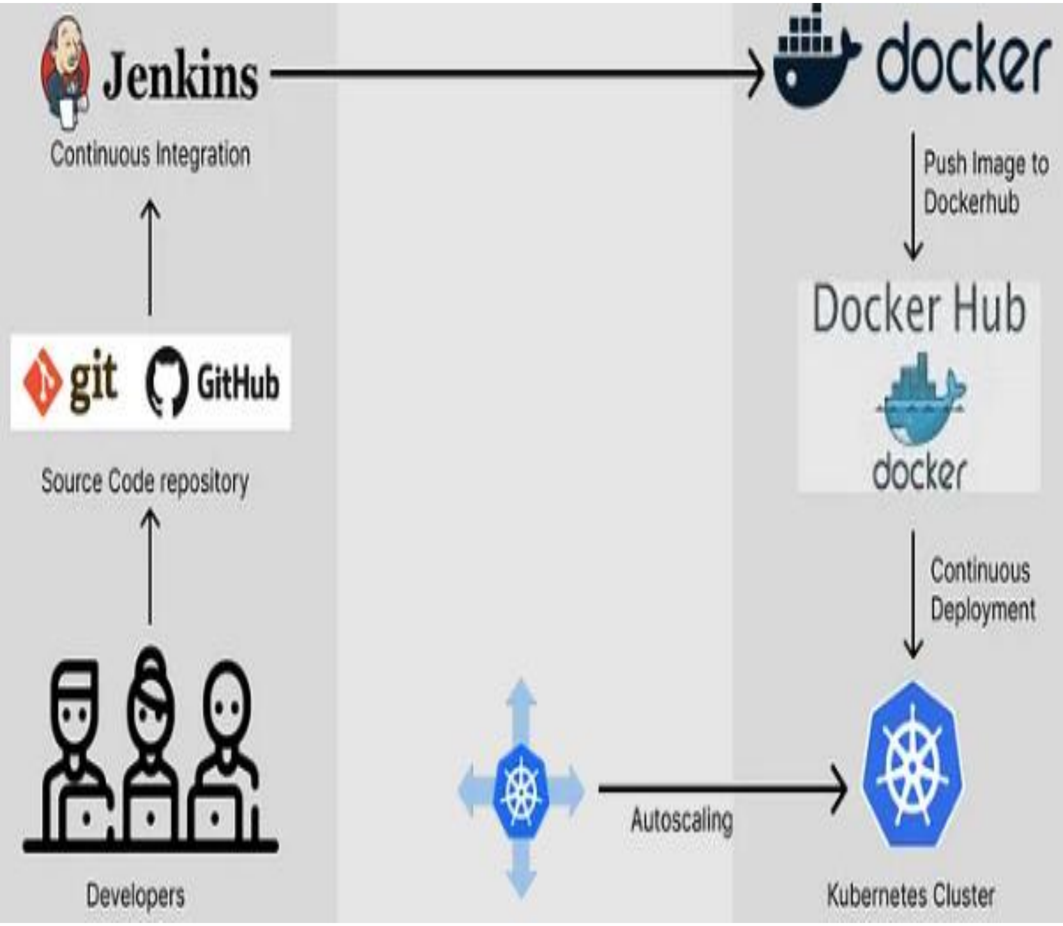
2. Jenkins for Continuous Integration

- **Jenkins**: A CI/CD tool that automates building, testing, and integrating code.
- **Process**:
  - Jenkins pulls code from the GitHub repository.
  - Jenkins builds the application and runs tests to ensure code quality.
  - Once successful, Jenkins triggers the next step (containerization with Docker).



3. Docker for Containerization

- **Docker**: Used to package the application and its dependencies into a portable container.
- **Steps**:
    - Jenkins uses Docker to create a **Docker image** of the application.
    - The image is then pushed to **Docker Hub**, a cloud-based repository for storing and sharing Docker images.



4. Continuous Deployment with Kubernetes

- **Kubernetes Cluster**: After the Docker image is available in Docker Hub, it is deployed to a Kubernetes cluster.
- **Key Features**:
    - Kubernetes handles **orchestration**, ensuring that containers are deployed across multiple nodes.
    - It enables **auto scaling**, meaning the number of containers increases or decreases based on demand.

CI/CD Pipeline:

Continuous Integration (CI):

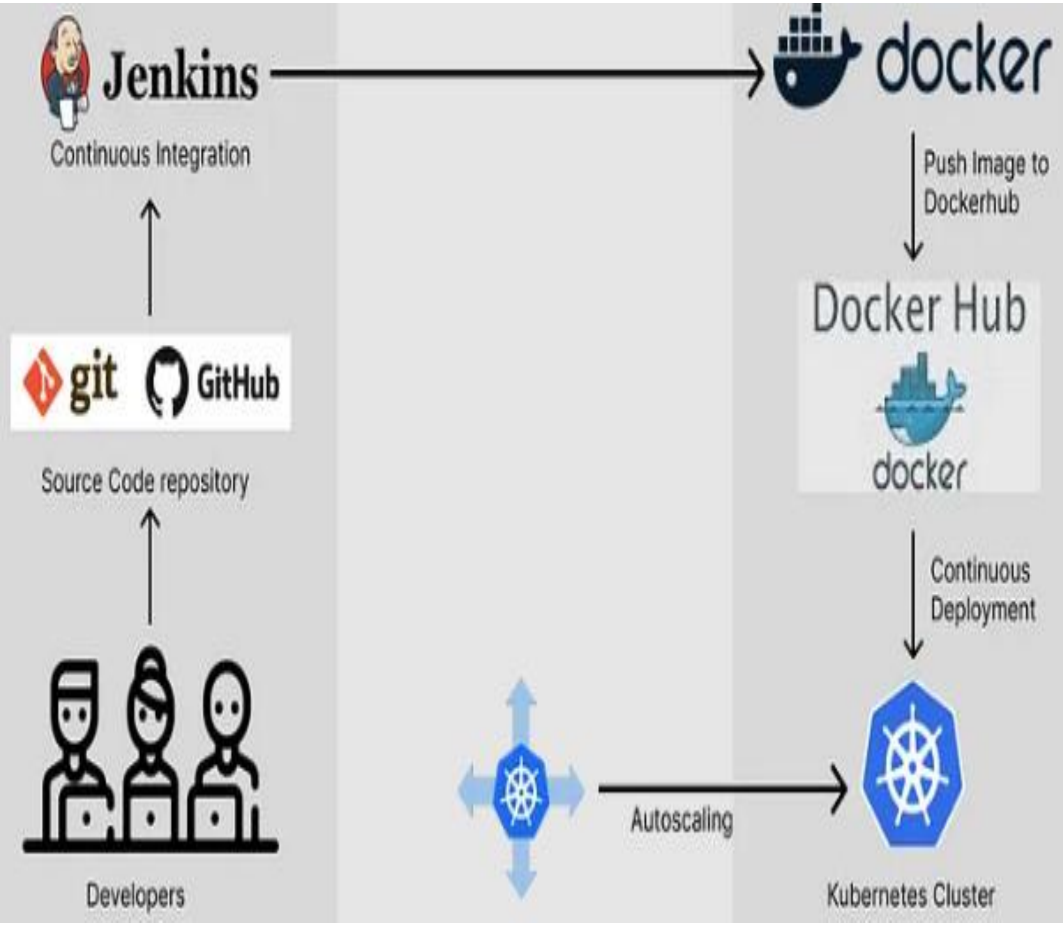
- Automates merging code changes into a shared repository.
- Runs automated tests to validate changes.
- Detects integration issues early.

Continuous Delivery (CD):

- Prepares code changes for release to production.
- Ensures the application is always in a deployable state.

Continuous Deployment:

- ensures that tested, stable changes are delivered to users quickly and reliably.
- Minimizes human intervention in deployment.



CI/CD Pipeline Workflow

CI/CD Pipeline Workflow

What does DevOps primarily focus on?

a) Only coding
b) Automation of deployment
c) Collaboration between development and operations teams
d) Monitoring software after deployment

What is the key purpose of CI (Continuous Integration)?

a) Automate deployment to production
b) Merge and test code changes frequently
c) Automate rollback processes
d) Enable manual code testing

In a CI/CD pipeline, which step comes immediately after “Build”?

a) Deployment
b) Integration tests
c) Code commit
d) Unit tests

- **Which of the following is NOT a DevOps tool?**

a) Jenkins
b) Kubernetes
c) Ansible
d) Adobe Photoshop

- **What is the primary difference between Continuous Delivery and Continuous Deployment?**

a) Continuous Delivery automates testing, while Continuous Deployment automates building
b) Continuous Delivery involves manual approval, while Continuous Deployment is fully automated
c) Continuous Deployment involves a staging environment, while Continuous Delivery does not
d) There is no difference

- **What is the main benefit of automating the CI/CD pipeline?**

a)Reduces server costs
b) Ensures faster and more reliable software releases
c) Eliminates all testing needs
d) Increases manual intervention during deployment

- **Which phase of the CI/CD pipeline is responsible for verifying interactions between software modules?**

a) Unit testing
b) Build
c) Integration testing
d) Staging